"AS-BUILT" SPECIFICATION FOR CCIT7 PROCESSOR PROGRAM

Job Orders 73-783 and 71-475

(TIRF 78-0022)

Prepared By

Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

SPACE AND LIFE SCIENCES DIRECTORATE

National Aeronautics and Space Administration

LYNDON B. JOHNSON SPACE CENTER

Houston, Texas

November 1978

LEC-12518

JSC-14554

"AS-BUILT" SPECIFICATION FOR CCIT7 PROCESSOR PROGRAM

Job Orders 73-783 and 71-475

(TIRF 78-0022)

PREPARED BY

W. P. White

APPROVED BY

James A Wilkinson

for P. L. Krumm, Acting Supervisor
Scientific Applications Software Section

Elmer M. Hsu

E. M. Hsu, Supervisor
Accuracy Assessment Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

Space and Life Sciences Directorate

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

November 1978

LEC-12518

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
JSC-14554		

4. Title and Subtitle	5. Report Date
"As-Built" Specification for CCIT7 Processor Program	November 1978
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
W. P. White Lockheed Electronics Company, Inc.	LEC-12518
	10. Work Unit No.

9. Performing Organization Name and Address	11. Contract or Grant No.
Lockheed Electronics Company, Inc. 1830 NASA Road 1 Houston, Texas 77058	NAS 9-15200
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Lyndon B. Johnson Space Center Houston, Texas 77058 Technical Monitor: J. D. Erickson	Technical Report
	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

The program CCIT7 is a utility module of the Accuracy Assessment Software System of the Large Area Crop Inventory Experiment. This program accesses data originating on the Classification and Mensuration Subsystem/Crop Assessment Subsystem interface tapes (CCIT's) of the Earth Resources Interactive Processing System (version 7). The data items needed for subsequent Accuracy Assessment processing are written into three disk files. The data extracted consist of a summary of the classification stratified areal estimate, cluster-dot match, and analyst-labeled dots (types 1 and 2).

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Accuracy Assessment CCIT	

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	65	

*For sale by the National Technical Information Service, Springfield, Virginia 22161

CONTENTS

TABLES

FIGURES

ABBREVIATIONS

AA	Accuracy Assessment
CAMS	Classification and Mensuration Subsystem
CAS	Crop Assessment Subsystem
CCIT	CAMS/CAS interface tape
DEC	Digital Equipment Corporation
DPR	Data processing request
DTL	Data Techniques Laboratory
DTRM	Data terminal
EOD	Earth Observations Division
ERIPS	Earth Resources Interactive Processing System
LACIE	Large Area Crop Inventory Experiment
Pixel	Picture element
SAE	Stratified areal estimate
TIRF	Transmittal Information Request Form
UIC	User identification code

1. SCOPE

This document specifies the detailed design for a software module called CCIT7, which will manipulate and extract data from Accuracy Assessment (AA) data base files. These files are derived from Large Area Crop Inventory Experiment (LACIE), version 7, Classification and Mensuration Subsystem/Crop Assessment Subsystem (CAMS/CAS) interface tapes (CCIT's). The data extracted are output into three new data base files for direct input to AA analytical programs.

2. APPLICABLE DOCUMENTS

The following documents, of the exact issue shown, form parts of the specification to the extent specified herein.

a. "As-Built" Design Specification for PDP 11/45 Accuracy Assessment System Using Disk Data File. JSC-13893 (LEC-11881), February 1978 (and references therein).

b. Implementation of CCIT6A Processor Program. Transmittal Information Request Form (TIRF) 78-0022, May 11, 1978.

c. CAM/CAS Interface Tape Interface Control Document. LACIE-C00708, revision A (JSC-09866), July 1976.

d. Classification and Mensuration Subsystem (CAMS) Requirements. LACIE-C00200, volume II, revision D (JSC-11330), August 1977.

e. "As-Built" Design Specification for CCIT6A Processor Program. JSC-14368 (LEC-12303), August 1978.

3. SYSTEM DESCRIPTION

The CCIT7 processor module accomplishes the data manipulations shown in figure 1. Basically, the CCIT data for a particular segment number, SSSS, and classification date, YYDDD, contained in file SSSSYYDDD.CCO are processed to obtain three output files required as input to existing or planned AA programs. The SSSSYYDDD.CLO file contains data needed for future programs. The SSSSYYDDD.AI1 and SSSSYYDDD.AI2 files are required for input to existing modules SPATL and MLTCRP.

3.1 HARDWARE DESCRIPTION

The PDP 11/45, with the following peripherals, is required.

a. Card reader or user terminal

b. Line printer

c. Two disk units

3.2 MODULE DESCRIPTION

The CCIT7 module is implemented on the PDP 11/45 for time-sharing or background processing of CCIT data files into three output data files: an unformatted file of character data and two formatted files of analyst-labeled dots. See the functional flow diagram (fig. 2). The CCIT7 module represents a major revision of the CCIT6A module described in reference e (section 2).

The LACIE CCIT is a universal nonimaging tape containing extensive statistical and ancillary data for a series of Earth Resources Interactive Processing System (ERIPS) runs. Using the AA CCIT program, all data for a relevant segment are transferred to a Files-11 disk file named SSSSYYDDD.CCO,···, where SSSS is the segment number, YY is the year, and DDD is the day of the year. This .CCO file contains three 80-byte header records and a large number (>20) of 720-byte data records.

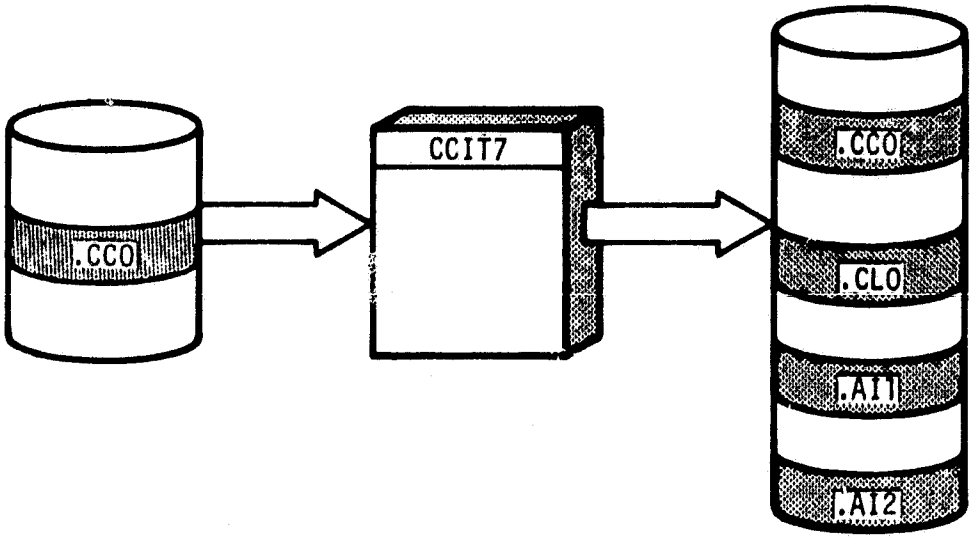

Figure 1.— Data flow of the CCIT7 processor program.

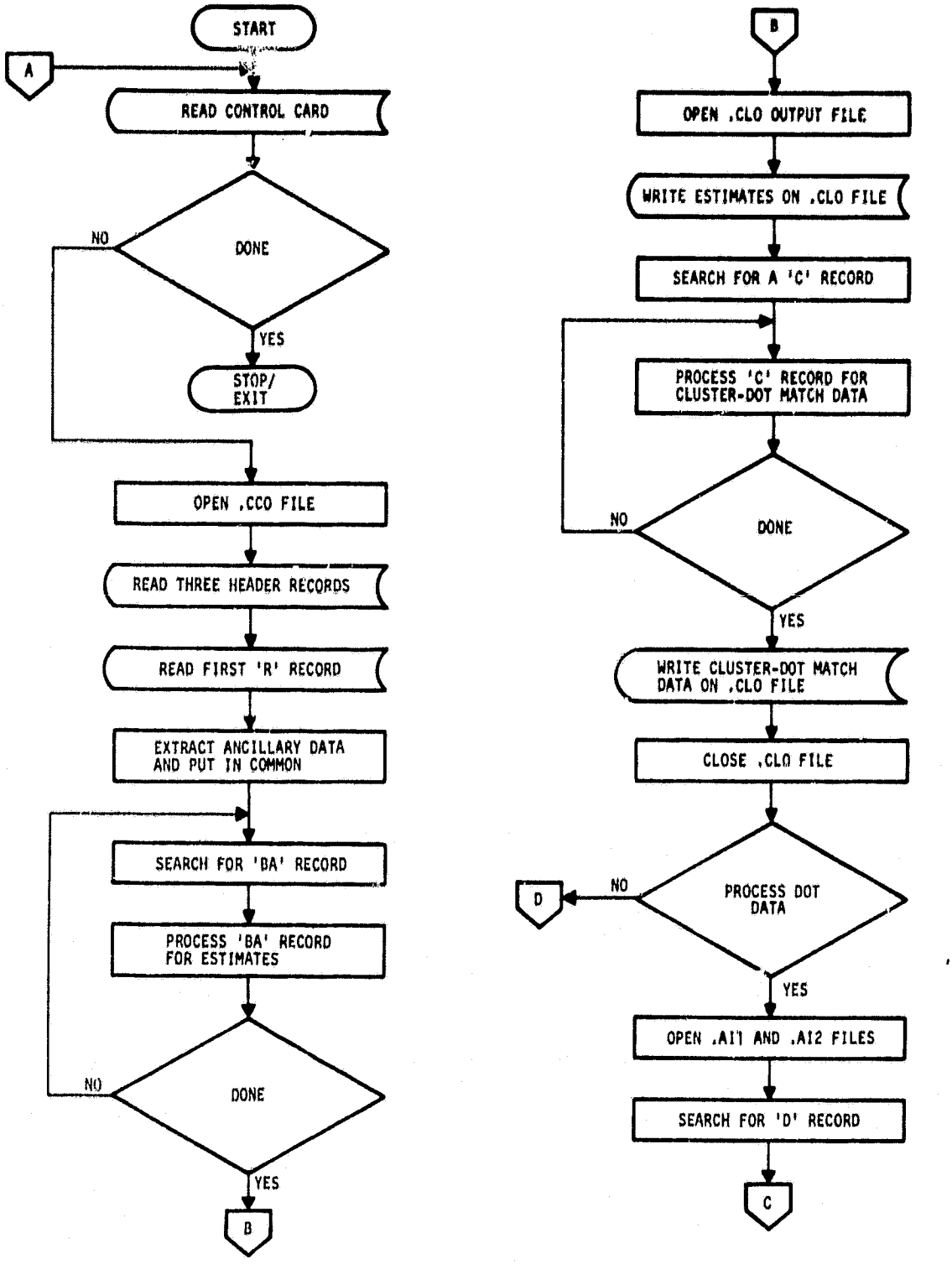

Figure 2.— Functional flow of the CCIT7 processor program.

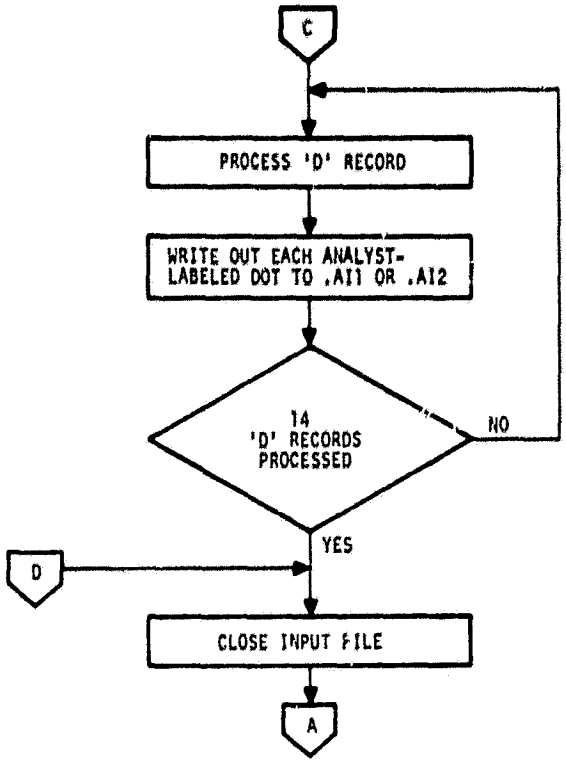

Figure 2.– Concluded.

The first step of the process is to read the name of the input .CCO file and open this file for reading. Then the three CCIT header records are read and ignored. The next record (720 bytes) is read, checked to verify that it is a recognition (R) record, and processed to extract the data processing request (DPR) number and acquisition dates used in the classification.

Next a 'BA' record is searched for and processed. The program extracts classification data on class picture element (pixel) populations, the ERIPS estimate, the stratified areal estimate (SAE), and the variance for each class. An output file SSSSYYDDD.CLO (where SSSS is the segment number and YYDDD is the classification date derived from the DPR number) is opened. The first record of the .CLO file contains an integer word giving the number of crop classes for the classification. The estimate data are written as the second record. Sixteen bytes are required for each class; a LACIE version 7 CCIT can contain data on up to 26 classes. A line printer report is output after the disk files are written.

In the next step a 'C' record is searched for and processed. The total number of clusters (Q) and cluster-dot match data are extracted as Q-groups of 12 characters. (Generally, there are more than 15 clusters, so some of these data appear in additional 'C' records.) When all the cluster-dot data are assembled into a buffer, the number of clusters is written out as the third record of file SSSSYYDDD.CLO, and the match data are written as the second record of this file. A line printer report is then output, and the SSSSYYDDD.CLO file is closed.

Then the program tests to determine if analyst-labeled dot output files are required; this is the default condition. If this condition exists, the output files SSSSYYDDD.AI1 and SSSSYYDDD.AI2 are opened, and a search is made for the first dot record. There are 14 dot records listing all 209 dots. The program examines each dot to determine if it has been labeled by the analyst. If so, it is written out (line, sample, and label) to the proper file, depending on the dot type (1 or 2). The first dot record in

each output file also contains ancillary information on the segment (number and state code), classification date, acquisition dates, data terminal (DTRM) tape number, and type of label.

When all dots have been processed, the SSSSYYDDD.AI1 and SSSSYYDDD.AI2 output files and the input file are closed. No report is output detailing the .AI file dot data because these formatted files are easily examined using the Digital Equipment Corporation (DEC) PDP-11 utility program PIP.

3.3 SOFTWARE DESCRIPTION

The CCIT7 processor program consists of 11 user-supplied routines: CCIT7 (main program), READH, READRC, HEADER, BIASC7, CLUST7, RITE7, TURNON, DOTS7, STCOD7, and PRINT7. The program makes use of a card-image-formatted file, CCIT7.DAT, for program control and the line printer and user disk for output. The following sections provide a detailed description of each of the 11 routines. The recommended task-build command file (CCIT7.CMD), used to create the load module (CCIT7.TSK), is given in table 1.

TABLE 1.- TASK-BUILDER COMMAND FILE FOR CCIT7 PROCESSOR PROGRAM

```
CCIT7,LP:/SH=CCIT7,READH,READRC,HEADER,BIASC7,CLUST7,
RITE7,TURNON,DOTS7,STCOD7,PRINT7
/
FMTBUF=132
UNITS=6
ACTFIL=6
ASG=SY:1
ASG=SY:2
ASG=SY:3
ASG=SY:5
ASG=LP:6
PRI=50
//
```

For simplicity, the definition of arrays carried in COMMON blocks, the definition of COMMON blocks, and the description of COMMON blocks are not repeated for each routine. Instead, each of these elements is described in the

routine of origin. Reference to the Interface subsections and to the compiler listings of each routine provides sufficient information to follow the data flow throughout the program.

3.3.1 MODULE CCIT7

3.3.1.1 Linkage

The CCIT7 program is the main program. It calls user subroutines READH, READRC, HEADER, BIASC7, CLUST7, DOTS7, and PRINT7. Subroutines BIASC7, DOTS7, and PRINT7 are called using multiple entry points.

3.3.1.2 Interface

Most communication with the user routines is handled via COMMON blocks. A single integer parameter is passed on call to READH, which indicates the number of CCIT header records to be read. A single integer parameter Q is passed to CLUST7 as a flag.

3.3.1.2.1 COMMON Block BUF

BUF contains a 720-byte array, A, which is used to hold one CCIT logical record for processing.

3.3.1.2.2 COMMON Block FNAME

FNAME contains a 24-byte array, FILNAM, and an integer variable, SKIP. FILNAM contains the input file name read from CCIT7.DAT. The value of SKIP determines whether the dot records are to be processed. If SKIP is nonzero, the dots are not processed.

3.3.1.2.3 COMMON Block B7

B7 contains the number of categories, NCAT, for the classification.

3.3.1.2.4 COMMON Block CLUSTR

CLUSTR contains a 60-byte by 12-byte array, CNAME, and an integer variable CNUM. CLUSTR provides an interface between subroutine CLUST7 and subroutine RITE7.

3.3.1.3 Input

The CCIT7 program receives all input CCIT data via subroutine READRC. Control data are provided directly by reading card images from file CCIT7.DAT.

3.3.1.4 Output

The CCIT7 program provides all output via subroutines PRINT7, RITE7, BIASC7, DOTS7, and CLUST7.

3.3.1.5 Storage

The CCIT7 program requires 1078 words of storage.

3.3.1.6 Description

The CCIT7 routine provides the control function for the program. Flow is controlled via tests on the first bytes (descriptive characters) of each logical record in the CCIT input file and by counts based upon the required number of records of a given type.

3.3.1.7 Flow Chart

The flow chart for CCIT7 is given in figure 3.

3.3.1.8 Listing

The listing for this subroutine is given in figure 4.

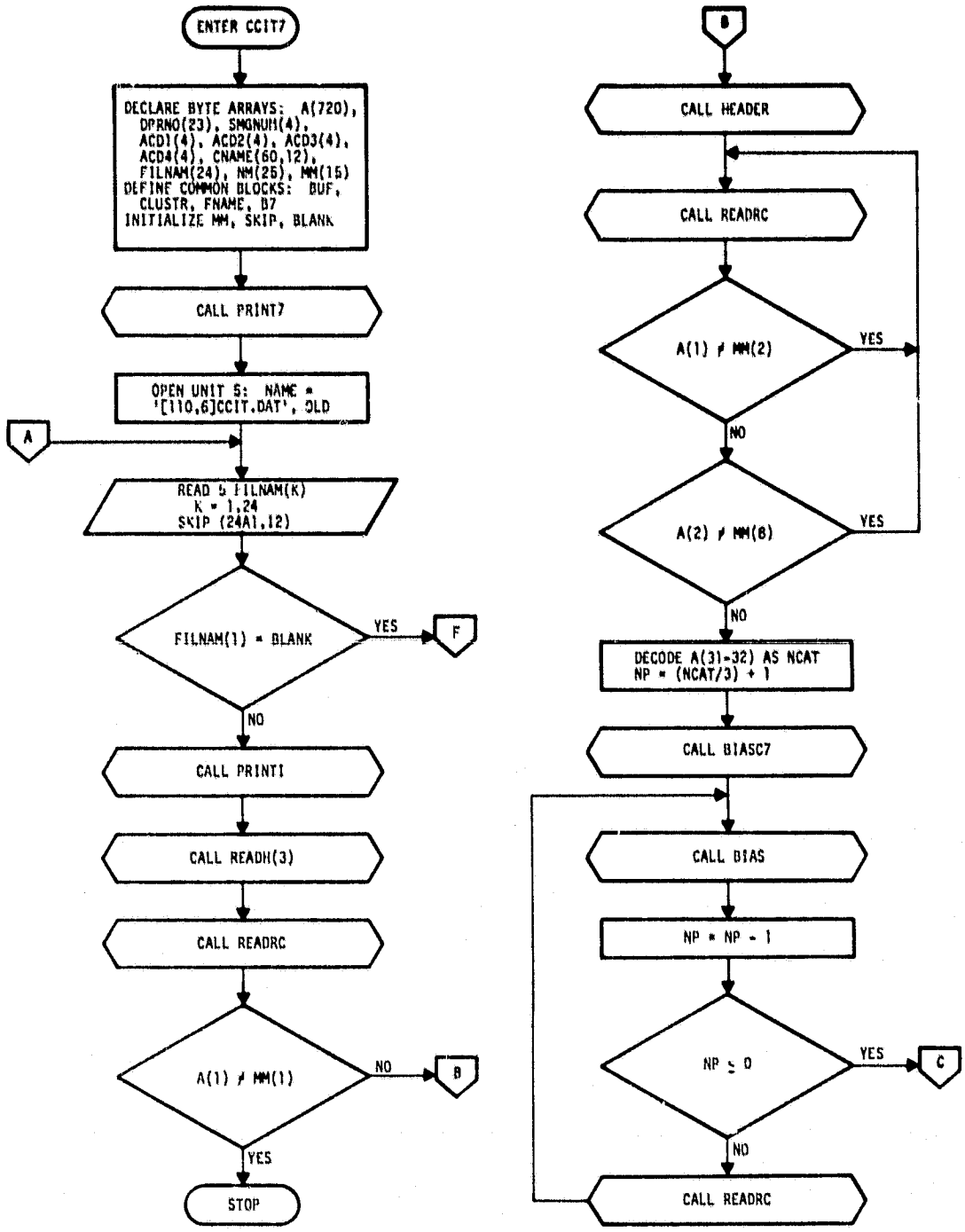

Figure 3.— Flow diagram for the CCIT7 processor program.

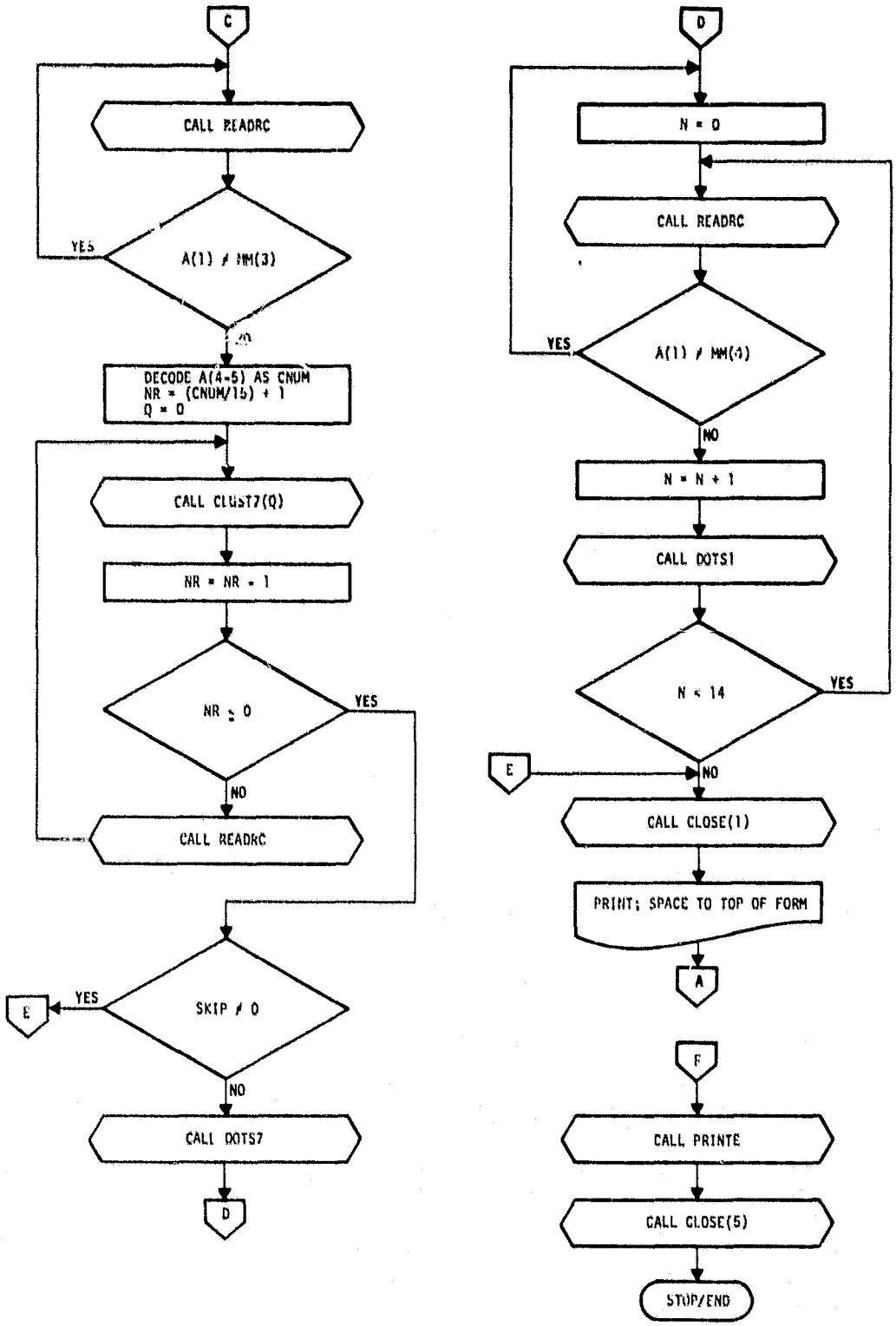

Figure 3.— Concluded.

```
FORTRAN IV-PLUS V02-51          12:43:06    17-JUL-78          PAGE 1
CCIT7.FTN       /TR:BLOCKS/WR

0001            PROGRAM CCIT7
0002            IMPLICIT INTEGER(A-Z)
0003            BYTE A(720),DPRNR(23),SHORUR(4),ACQS(4),ACQB(4),ACQJ(4),RCD4(4),
               1  CNAME(60,12),FILNAM(24),NP(25),PM(15),BLANK
                C**************************************************************
                C       MAIN OF CCIT7 PROCESSOR - READ A CARD IMAGES FROM [110,6]CCIT7.DAT
                C       WHICH GIVES THE FILE NAME OF THE LACIE7 AA CCIT FILE ON DISC
                C       IN THE FORM: [131,7]SSSSSYYDDD,CCO
                C       A(720) IS THE BUFFER WHICH CONTAINS ONE CCIT7 DATA RECORD
                C**************************************************************
0004            COMMON/BUF/A
0005            COMMON/B7/NCAT
0006            COMMON/CLUSTR/CNAME,CNUM
0007            COMMON/FNAME/FILNAM,SKIP
0008            DATA MM/1HR,1HS,1HC,1HD,1H0,1H1,1H2,1H4,7*1H /
0009            DATA SKIP/0/
0010            DATA BLANK/1H /
0011            CALL PRINT7
                C       READ INPUT FILNAME AND SKIP FACTOR
0012            OPEN(UNIT=9,NAME='[110,6]CCIT7.CAT',TYPE='OLD',READONLY,
               1  ACCESS='SEQUENTIAL',FORM='FORMATTED')
0013      800   READ(9,101)(FILNAM(K),K=1,24),SKIP
0014            IF(FILNAM(1).EQ.BL NK) GO TO 200
0015            CALL PRINT7
                C       READ 3 HEADER RECORDS OF 80 BYTES EACH
0016            CALL READH(3)
                C       READ THE FIRST 'R' RECORD AND PROCESS FOR DPR NUMBER,ACQUISITION
                C       DATES ETC
0017            CALL READRC
0018            IF(A(1).NE.MM(1)) STOP
0019            CALL HEADER
                C       READ AND PROCESS BIAS CORRECTION SUMMARY RECORD
                C       AS FIRST RECORD OF FILE SSSSYYDDD.CLO
0020      210   CALL READRC
0021            IF(A(1).NE.MM(2)) GO TO 210
0022            IF(A(2).NE.MM(8)) GO TO 210
0023            DECODE(2,100,A(3))NCAT
0024            NP = (NCAT/3) + 1
0025            CALL BIASC7
0026      245   CALL BIAS
0027            NP = NP - 1
0028            IF(NP.LE.0) GO TO 220
0029            CALL READRC
0030            GO TO 245
                C       READ FIRST CLUSTER RECORD
                C       WHEN DONE SUBROUTINE 'WRITE7' IS CALLED WHICH WRITES THE NUMBER
                C       OF CLUSTERS; AND THE CLUSTER-BY MATCH DATA OUT AS TWO RECORDS
                C       OF FILE SSSSYYDDD.'LO
0031      220   CALL READRC
0032            IF(A(1).NE.MM(3)) GO TO 220
0033            DECODE(2,100,A(4))CNUM
0034            NR = (CNUM/15) + 1
0035            C = 0
0036      235   CALL CLUST7(C)
0037            NR = NR -1
0038            IF(NR.LE.0) GO TO 225
```

Figure 4.— Listing for the CCIT7 processor program.

```
FORTRAN IV-PLUS V02-51          12143104    17-JUL-78          PAGE 2
CCIT7.FTN       /TR:BLOCKS/WR

0039            CALL READRC
0040            GO TO 235
      C         IF DBT DATA IS NOT NEEDED SKIP TO END OF PROGRAM
0041     225    IF(SKIP.NE.0) GO TO 250
      C         INITIALIZE THE DBT OUTPUT FILES AND COUNTERS
0042            CALL DBTS7
      C         READ DBTS RECORDS
0043     230    N=0
      C         LOOK FOR 'D' (209 DBTS) RECORDS
0044     240    CALL READRC
0045            IF(A(1).NE.MM(4)) GO TO 230
0046            N=N+1
      C         PROCESS A 'D' RECORD
0047            CALL DBTS1
0048            IF(N.LT.14) GO TO 240
      C         CLOSE INPUT FILE
0049     250    CALL CLOSE(1)
0050            PRINT 102
0051            GO TO 800
0052     260    CALL PRINTE
0053            CALL CLOSE(5)
0054            STOP
0055     100    FORMAT(I2)
0056     101    FORMAT(24A1,I2)
0057     102    FORMAT(1H1)
0058            END
```

14 3-12

Figure 4.— Concluded.

3.3.2 SUBROUTINE READH

3.3.2.1 Linkage

Subroutine READH calls subroutine TURNON.

3.3.2.2 Interface

READH interfaces with TURNON via an integer parameter (passed on call) giving the logical unit number to be opened and via COMMON block NAME containing the name of the file to be opened. COMMON block FNAME (see section 3.3.1.2.2) interfaces CCIT7 with READH. COMMON block BUF provides no true interfacing function for this routine.

3.3.2.2.1 COMMON Block NAME

NAME contains a 25-byte array, NM, which contains the complete name of a file to be opened by subroutine TURNON. NAME also interfaces several subroutines with subroutine PRINT7. The last byte of array NM should contain the null (0) character.

3.3.2.3 Input

Header records from the CCIT input file are input.

3.3.2.4 Output

The only output is a read error message to the line printer.

3.3.2.5 Storage

READH requires 504 words of storage.

3.3.2.6 Description

Subroutine READH spaces past the three 80-byte CCIT header records, and the CCIT file name is written into the NM array. Subroutine TURNON opens the file on unit 1, the three records are read, and READH returns to CCIT7.

3.3.2.7 Flow Chart

The flow diagram for subroutine READH is given in figure 5.

3.3.2.8 Listing

The listing for this subroutine is given in figure 6.

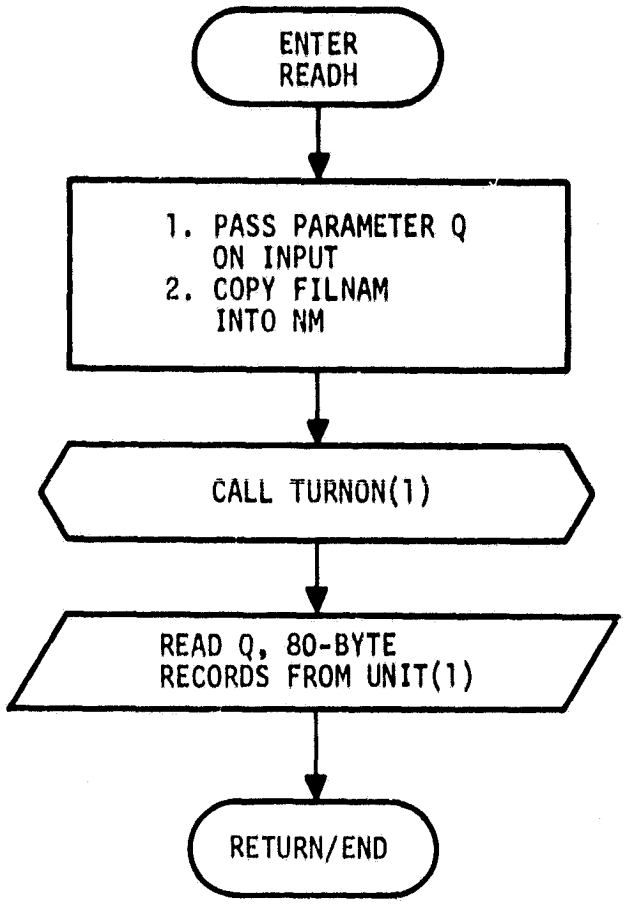

Figure 5.— Flow diagram for subroutine READH.

```
FORTRAN IV-PLUS V02-51          07:44:33    05-MAY-78          PAGE 1
READH.FTN       /TR:10:0C:3/WR

0001            SUBROUTINE READH(J)
0002            IMPLICIT INTEGER(A-Z)
                C
                C READ CCIT HEADER RECORDS FROM DISC FOR FILE FILNAM
                C
0003            BYTE NM(25),FILNAM(24),A(720)
0004            COMMON/BUFF/A
0005            COMMON/NAME/NM
0006            COMMON/FNAME/FILNAM,SKIP
0007            DO 10 JJ=1,24
0008      10    NM(JJ) = FILNAM(JJ)
0009            NM(25) = 0
0010            CALL TVR(NM(1))
0011            DO 1 J=1,8
0012      1     READ(1,ERR=99,END=9999)(A(K),K=1,720)
0013            RETURN
0014      99    PRINT 200,J
0015      200   FORMAT(1H ,' READ ERROR IN READH; ITERATION ',I2,//)
0016            STOP
0017            END
```

3-16
18

Figure 6.— Listing for subroutine READH.

3.3.3 SUBROUTINE READRC

3.3.3.1 Linkage

Subroutine READRC is called by CCIT7.

3.3.3.2 Interface

Subroutine READRC interfaces with CCIT7 via COMMON block BUF (see section 3.3.1.2.1).

3.3.3.3 Input

One data record read from the CCIT disk file is input.

3.3.3.4 Output

A read operation error message is output to the line printer.

3.3.3.5 Storage

This subroutine requires 441 words of storage.

3.3.3.6 Description

READRC reads one 720-byte logical data record from the CCIT input file into a buffer array, A.

3.3.3.7 Flow Chart

The flow diagram for subroutine READRC is given in figure 7.

3.3.3.8 Listing

The listing for this subroutine is given in figure 8.

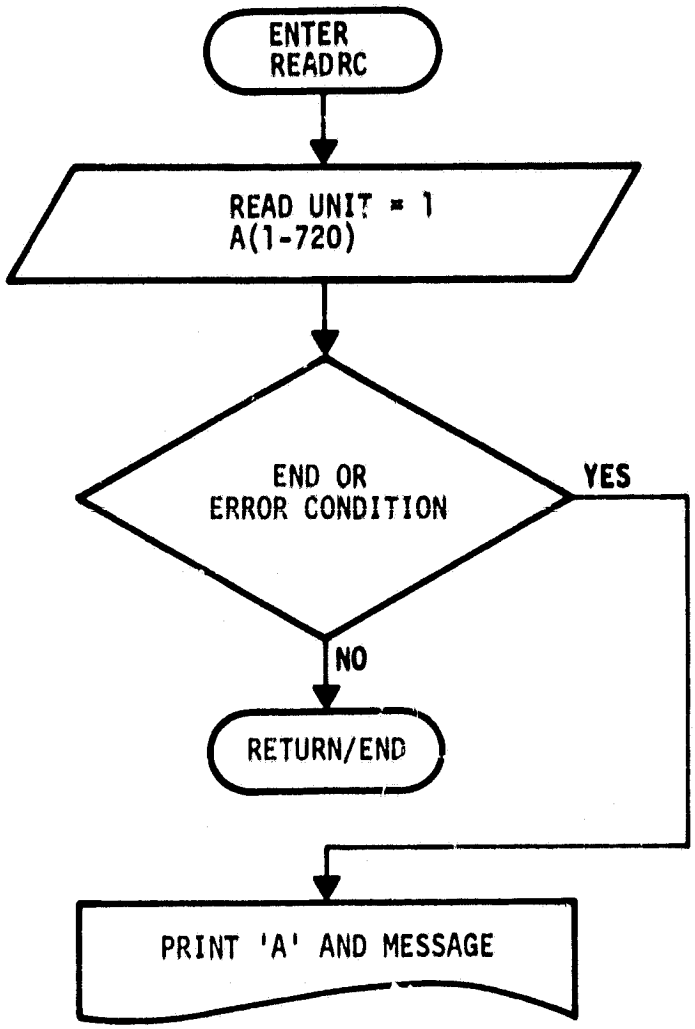

Figure 7.— Flow diagram for subroutine READRC.

```
FORTRAN IV-PLUS V02-51          07144149    05-MAY-78          PAGE 1
READRC.FTN      /TRACEBACKS/WR

0001            SUBROUTINE READRC
0002            IMPLICIT INTEGER(A-Z)
              C
              C READ A CCIT DATA RECORD OF 720 BYTES INTO THE BUFFER A
0003            BYTE A(720)
0004            COMMON/BUFF/A
0005            READ(1,ERR=99,END=99)(A(K),K=1,720)
0006            RETURN
0007         99 PRINT 100,A
0008        100 FORMAT(1H ,'ERROR IN READRC',///,' BUFFER CONTAINS',///,
              *1H ,8(120A1))
0009            END
```

3-79
31

Figure 8.— Listing for subroutine READRC.

3.3.4 SUBROUTINE HEADER

3.3.4.1 Linkage

Subroutine HEADER is called by CCIT7.

3.3.4.2 Interface

HEADER interfaces with CCIT7 via COMMON blocks BUF (see section 3.3.1.2.1) and DOTS and interfaces with PRINT7 (entry PRINTH) via COMMON block DOTS.

3.3.4.2.1 COMMON Block DOTS

DOTS contains a 23-byte array (DPRNO) that is used to hold the ERIPS DPR number, four 4-byte arrays (ACD1, ACD2, ACD3, and ACD4) that are used to store the acquisition dates used for the ERIPS run, and a 4-byte array (SMGNUM) that is used to store the LACIE segment number.

3.3.4.3 Input

There is no input to this subroutine.

3.3.4.4 Output

HEADER has no output.

3.3.4.5 Storage

This subroutine requires 445 words of storage.

3.3.4.6 Description

Subroutine HEADER selects byte data from the 'R' record of a CCIT (contained in buffer array A) and stores it into arrays in COMMON block DOTS. The data selected are the LACIE segment number [SMGNUM(1-4)], acquisition dates used for the ERIPS run [ACD1,ACD2,ACD3,ACD4], and ERIPS DPR number [DPRNO(1-23)]. The DPR number is printed in a message via a call to entry PRINTH of PRINT7.

3.3.4.7 Flow Chart

The flow diagram for subroutine HEADER is given in figure 9.

3.3.4.8 Listing

The listing for this subroutine is given in figure 10.

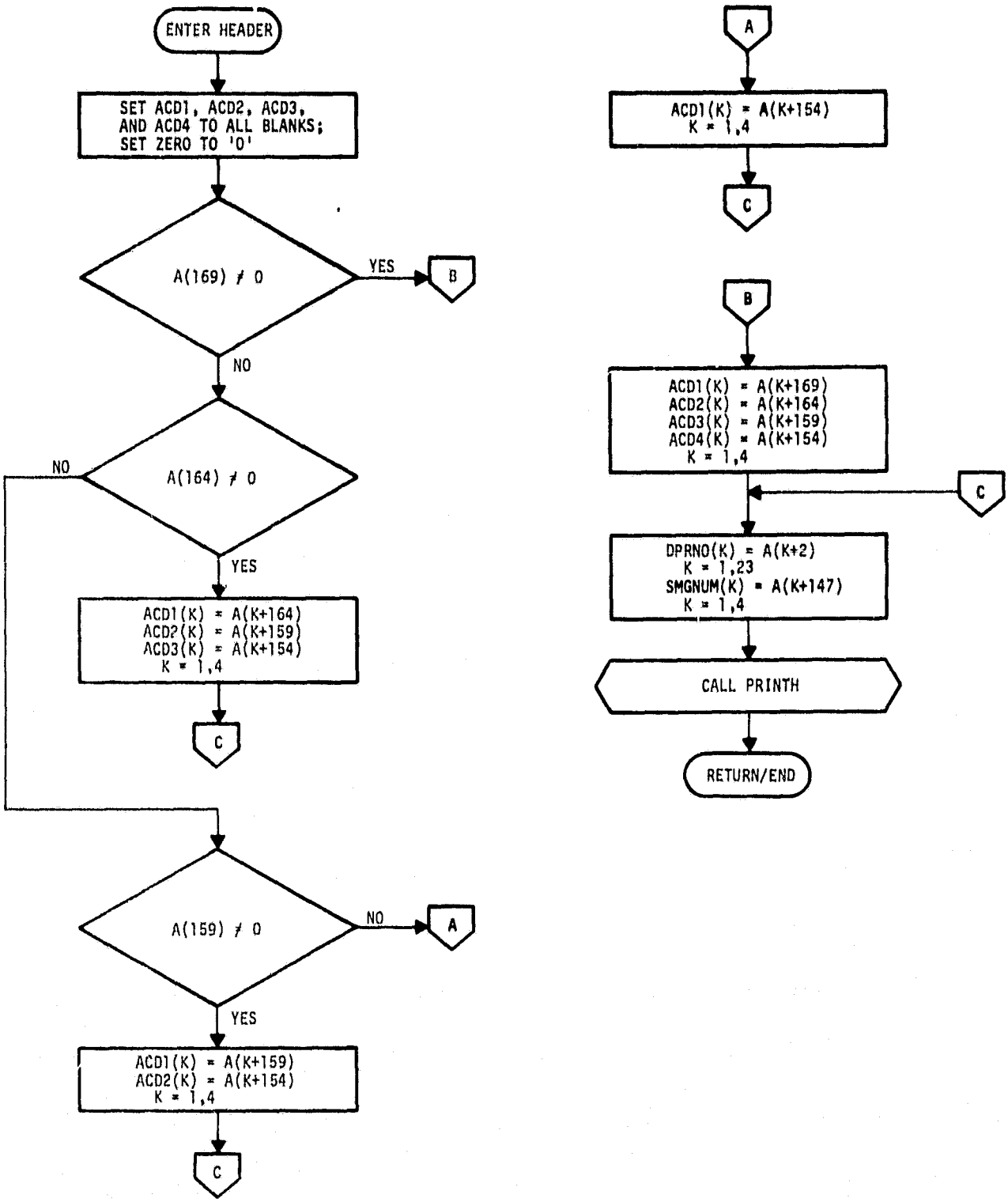

Figure 9.— Flow diagram for subroutine HEADER.

```
FORTRAN IV-PLUS V02-51          14:28:29    04-JUL-78          PAGE 1
HEADER.FTN      /TR:BLOCKS/WR

0001            SUBROUTINE HEADER
                C*************************************************************
                C     THIS SUBROUTINE OBTAINS THE DPR NUMBER ,AND ACQUISITION
                C     DATES FROM AN 'R' RECORD.
                C     THE ACQUISITION DATES ARE ORDERED BY LATEST DATE FIRST
                C     TO CONFORM TO OTHER AA SOFTWARE MODULE CONVENTIONS
                C*************************************************************
0002            IMPLICIT INTEGER(A-Z)
0003            BYTE A(720),ACD1(4),ACD2(4),ACD3(4),ACD4(4),
               1 DPRNB(23),SMGNUM(4)
0004            COMMON/BUF/A
0005            COMMON/DATS/DPRNB,ACD1,ACD2,ACD3,ACD4,SMGNUM
0006            DATA ACD1/4*1H /ACD2/4*1H /ACD3/4*1H /ACD4/4*1H /
0007            DATA ZERO/1H0/
                C     CHECK TO SEE HOW MANY ACQUISITIONS WERE USED
                C     ASSUMED TO BE IN THE ORDER OF EARLIEST FIRST
                C     WITH ASCII ZERO FILL
0008            IF(A(169).NE.ZERO) GO TO 10
0009            IF(A(164).NE.ZERO) GO TO 20
0010            IF(A(159).NE.ZERO) GO TO 30
0011            DO 4 K=1,4
0012       4    ACD1(K) = A(K+154)
0013            GO TO 40
0014      30    DO 5 K=1,4
0015            ACD1(K) = A(K+159)
0016       5    ACD2(K) = A(K+154)
0017            GO TO 40
0018      20    DO 6 K=1,4
0019            ACD1(K) = A(K+164)
0020            ACD2(K) = A(K+159)
0021       6    ACD3(K) = A(K+154)
0022            GO TO 40
0023      10    DO 7 K=1,4
0024            ACD1(K) = A(K+169)
0025            ACD2(K) = A(K+164)
0026            ACD3(K) = A(K+159)
0027       7    ACD4(K) = A(K+154)
                C     WRITE THE DPR NUMBER
0028      40    DO 3 K=1,23
0029       3    DPRNB(K) = A(K+2)
                C     WRITE SEGMENT NUMBER
0030            DO 1 K=1,4
0031       1    SMGNUM(K) = A(K+149)
                C     PRINT MESSAGE ON PRINTER
0032            CALL PRINTH
0033            RETURN
0034            END
```

Figure 10.— Listing for subroutine HEADER.

3.3.5 SUBROUTINE BIASC7

3.3.5.1 Linkage

BIASC7 is called once by the CCIT7 program. It calls subroutine TURNON once. An additional entry point, BIAS, is called one or more times by CCIT7.

3.3.5.2 Interface

BIASC interfaces with CCIT7 via COMMON blocks BUF (see section 3.3.1.2.1) and B7 (see section 3.3.1.2.3), and with TURNON via COMMON block NAME (see section 3.3.2.2.1).

3.3.5.3 Input

There is no input to this subroutine.

3.3.5.4 Output

BIASC7 writes two unformatted records onto unit 3. The data contained in this record are detailed in the appendix.

3.3.5.5 Storage

This subroutine requires 837 words of storage.

3.3.5.6 Description

BIASC7 codes the output file name as SSSSYYDDD.CLO, where SSSS is the segment number and YYDDD is the classification date. Unit 3 is opened for output via a call to subroutine TURNON. Then a record containing the integer NCAT (the number of categories or classes) is written on unit 3, and 12 times NCAT bytes of character data are output as the second record.

3.3.5.7 Flow Chart

The flow diagram for subroutine BIASC7 is given in figure 11.

3.3.5.8 Listing

The listing for this subroutine is given in figure 12.

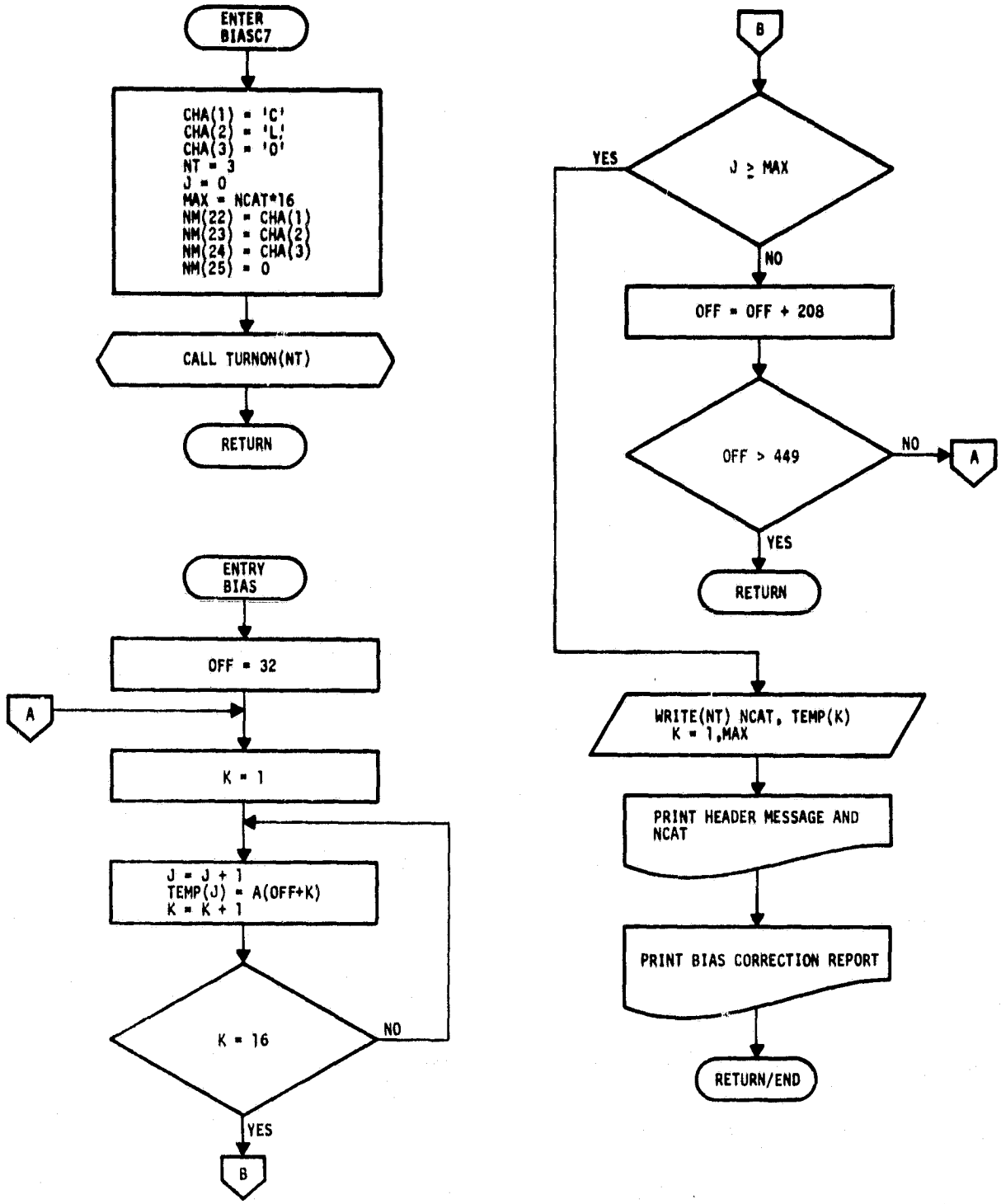

Figure 11.— Flow diagram for subroutine BIASC7.

```
FORTRAN IV-PLUS V02-51          10:19:25    25-JUL-78          PAGE 1
BIASC7.FTN       /I4:BLOCKS/X4

     1          SUBROUTINE BIASC7
0001            IMPLICIT INTEGER(A-Z)
0002       C*****************************************************************
           C       USES THE BIAS CORRECTION (I.E., STRATIFIED AREAL
           C       ESTIMATE DATA FROM A CCIT 'BA' RECORD
           C       TO FORM THE FIRST RECORD OF THE
           C       SSSSYYDDD.CLO OUTPUT FILE, LEAVES FILE OPEN FOR
           C       WRITING OF THE MORE RECORDS; NUMBER OF CLUSTERS,
           C       AND THE ALPHANUMERIC CLUSTER-CAT MATCH TABLE
           C       BY SUBROUTINE CLUST7
           C*****************************************************************
0003                COMMON/B77/NCAT
0004            BYTE NM(25),SMGNUM(4),OPRN2(23),ACD1(4),ACD2(4),ACD3(4),
                1   ACD4(4),CHA(3),TEMP(416)
0005            BYTE A(720)
0006                COMMON/BUFFA
0007                COMMON/NAME/NM
0008                COMMON/OPTS/OPRN2,ACD1,ACD2,ACD3,ACD4,SMGNUM
0009            DATA CHA/1HC,1HL,1HO/
           C*****************************************************************
           C       THIS PART OPENS OUTPUT FILE AND SETS UP THE
           C       PROCESSING ONLY -- ENTRY BIAS IS CALLED ONCE
           C       FOR EACH 'BA' RECORD PROCESSED
           C*****************************************************************
0010            NT = 3
0011            J = 0
0012            MAX = NCAT*15
           C       CODE NAME FOR OUTPUT FILE IN FORM SSSSYYDDD.CLO
0013            NM(22) = CHA(1)
0014            NM(23) = CHA(2)
0015            NM(24) = CHA(3)
0016            NM(25) = 0
           C       OPEN OUTPUT FILE = UNFORMATTED
0017            CALL TURBO(-1)
0018            RETURN
           C*****************************************************************
           C       ENTER HERE ONCE FOR  EACH 'BA' RECORD PROCESSED.
           C       ONCE ALL CLASSES (*CATAGORIES*) HAVE BEEN WRITTEN
           CC      INTO BUFFER TEMP.  RITE TO DISK AND PRINT REPORT
           C*****************************************************************
0019            ENTRY BIAS
0020            OFF = 32
0021       2    DO 1 K=1,16
0022            J = J + 1
0023       1    TEMP(J) = A(OFF+K)
0024            IF(J.GE.MAX) GO TO 10
0025            OFF = OFF + 208
0026            IF(OFF.GT.449) RETURN
0027            GO TO 2
           C*****************************************************************
           C       WRITE DISK RECORD AND PRINTER REPORT
           C*****************************************************************
0028       10   WRITE(NT)NCAT
0029            WRITE(NT)(TEMP(K),K=1,MAX)
0030            WRITE(6,200)NCAT
```

Figure 12.— Listing for subroutine BIASC7.

```
FORTRAN IV+PLUS V02-51          TUE19125    29-JUL-78          PAGE 2
DI&SC7.FTN      /TR:BL@CKS/mR

0031                WRITE(6,300)(TEMP(K),K=1,MAX)
0032                RETURN
0033          200   FORMAT(1H ,15X,'BIAS CORRECTION ',//,15X,'FOR ',I2,' CATAGORIES')
0034          300   FORMAT(1H ,26(5X,'CATAGORY LABEL ',A1,' POPULATION ',9A1,//,
                1   10X,'UNCORRECTED ''1A1,',',2A1,' CORRECTED ',1A1,',',2A1,
                2   ' VARIANCE ',2A1,''',2A1,///))
0035                END
```

Figure 12.— Concluded.

3.3.6 SUBROUTINE CLUST7

3.3.6.1 Linkage

Subroutine CLUST7 is called one to four times by CCIT7. CLUST7 calls RITE7 for data output and calls the PRINTC entry to PRINT7 for an output message.

3.3.6.2 Interface

CLUST7 interfaces with CCIT7 via COMMON block BUF (see section 3.3.1.2.1) and passed parameter Q, with RITE7 via COMMON block CLUSTR (see section 3.3.1.2.4), and with PRINT7 via passing parameter RCNUM on call.

3.3.6.3 Input

There is no input to this subroutine.

3.3.6.4 Output

Subroutine CLUST7 has no output.

3.3.6.5 Storage

This subroutine requires 836 words of storage.

3.3.6.6 Description

CLUST7 processes the CCIT 'C' records to provide the total number of clusters and the identity of the analyst-labeled (type 1) dot used to name each cluster.

On the first record processed, CLUST7 decodes bytes 4 and 5 to obtain the total number of clusters (CNUM) and bytes 6 and 7 to obtain the number of clusters contained on the record [RCNUM(1-15)]. Then for each cluster, the 12 bytes representing the cluster name (6 bytes) and dot name (6 bytes) used in labeling the cluster are copied into the array CNAME. PRINT7 is called via entry PRINTC to print a message containing the parameter RCNUM. When CNUM sets of data are written into CNAME, CLUST7 calls subroutine RITE7 for output.

3.3.6.7 Flow Chart

The flow diagram for subroutine CLUST7 is given in figure 13.

3.3.6.8 Listing

The listing for this subroutine is given figure 14.

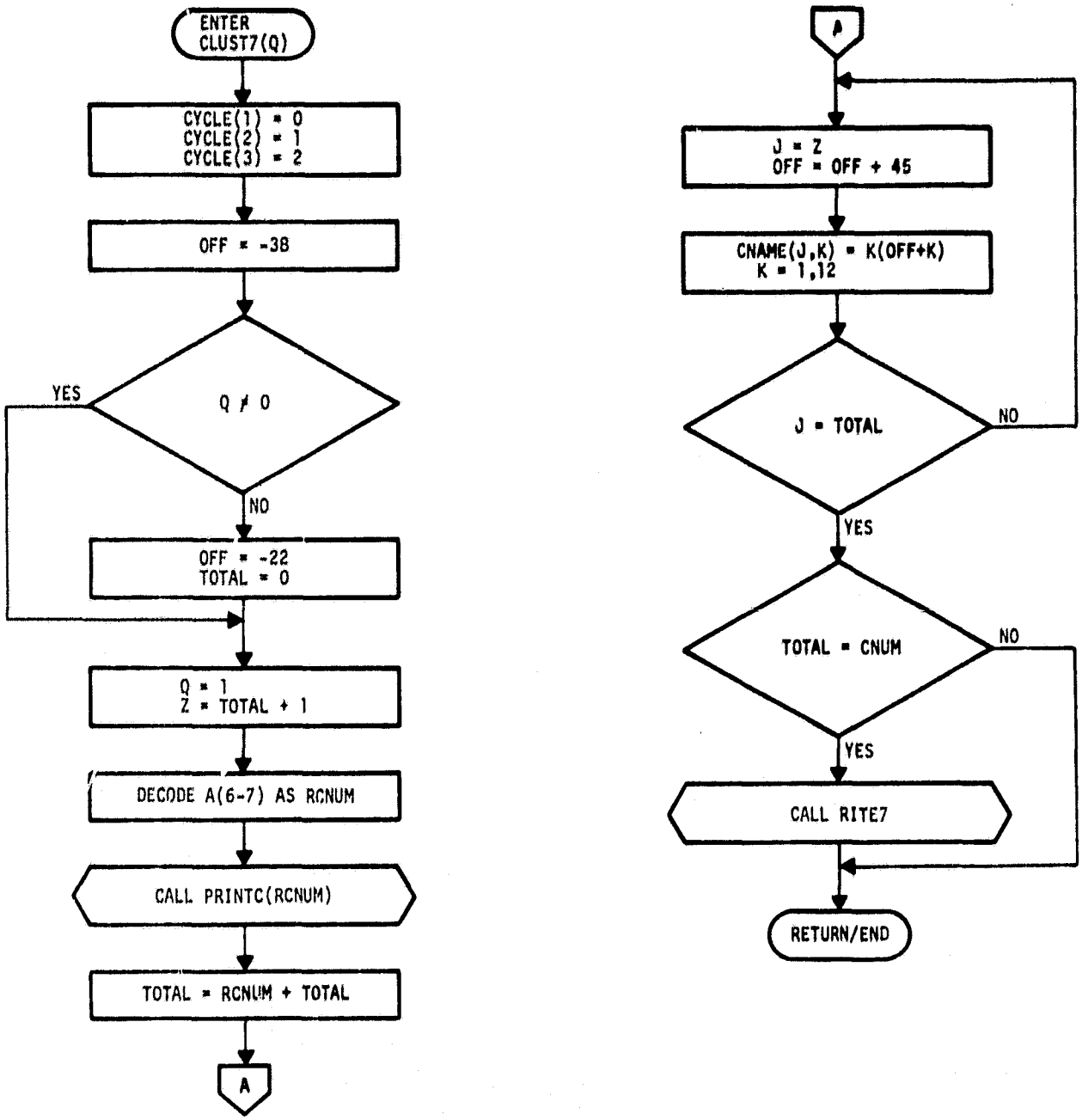

Figure 13.— Flow diagram for subroutine CLUST7.

```
FORTRAN IV-PLUS V02-51          14:28:48    06-JUL-78          PAGE 1
CLUST7.FTN      /TRIBLOCKS/WR

0001            SUBROUTINE CLUST7(G)
0002            IMPLICIT INTEGER(A-Z)
          C     READS CCIT 'C' RECORDS AND OUTPUTS CLUSTER LABEL AND CLUSTER-
          C     DBT MATCH DATA TO " DISC FILE NAMED SSSSYYDDD.CLG,
0003            BYTE A(720),CYCLE(3),CNAME(60,12)
          C.....................*.....................................
          C     CNAME IS A STORAGE ARRAY TO PERMIT WRITING THE DATA IN ONE
          C     RECORD TO DISC IN A FILE NAMED SSSSYYDDD.CLG
          C     CNUM IS THE TOTAL NUMBER OF CLUSTERS
          C     OFF IS THE BYTE OFFSET WITHIN A 'C' RECORD
          C     RCNUM IS THE NO. OF CLUSTERS IN THE CURRENT RECORD
          C.....................*.....................................
0004            COMMON/BUF/A
0005            COMMON/CLUSTR/CNAME,CNUM
0006            DATA CYCLE/1H0,1H1,1H2/
0007            OFF = -38
0008            IF(G.NE.0) GO TO 5
0009            OFF = -22
0010            TOTAL = 0
0011       5    G = 1
0012            I = TOTAL + 1
0013            DECODE(2,100,A(G))RCNUM
0014            CALL PRINTC(RCNUM)
0015            TOTAL = RCNUM + TOTAL
0016            DO 1 J=I,TOTAL
0017            OFF = OFF + 45
0018            DO 1 K=1,12
0019       1    CNAME(J,K) = A(OFF+K)
          C
          C     CHECK IF DONE
          C
          C     CALL RITE3N FOR DISK WRITE AND PRINTED REPORT
          C
0020            IF(TOTAL.EQ.CNUM) CALL RITE7
0021            RETURN
0022     100    FORMAT(I2)
0023            END
```

Figure 14.— Listing for subroutine CLUST7.

3-81
33

3.3.7 SUBROUTINE RITE7

3.3.7.1 Linkage

Subroutine RITE7 is called once by subroutine CLUST7.

3.3.7.2 Interface

Subroutine RITE7 interfaces with CLUST7 via COMMON block CLUSTR (see section 3.3.1.2.4).

3.3.7.3 Input

There is no input to RITE7.

3.3.7.4 Output

RITE7 writes two records onto a previously opened file (unit 3). This unit is opened in subroutine BIASC7 as an unformatted FORTRAN disk file. In addition, RITE7 writes a line printer report of the cluster-dot match data for AA evaluation.

3.3.7.5 Storage

This subroutine requires 484 words of storage.

3.3.7.6 Description

RITE7 writes two records onto unit 3. The first record is a single integer, CNUM. The second record consists of the array CNAME as CNUM 12-byte elements. The output file is closed via a call to the system routine CLOSE.

3.3.7.7 Flow Chart

The flow diagram for subroutine RITE7 is given in figure 15.

3.3.7.8 Listing

The listing for this subroutine is given in figure 16.

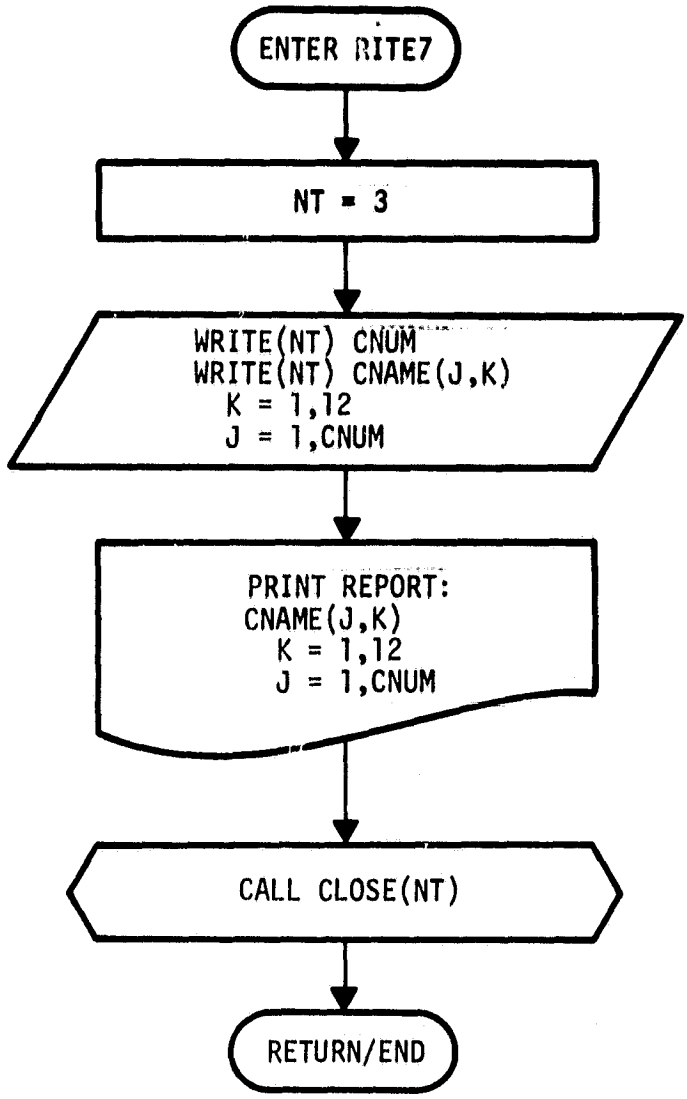

Figure 15.— Flow diagram for subroutine RITE7.

```
FORTRAN IV-PLUS V02-51          14112144    30-JUN-78          PAGE 1
RITE7.FTN       /TR:BLOCKS/-T

0001            SUBROUTINE RITE7
0002            IMPLICIT INTEGER(A-Z)
        C       WRITES AN UNFORMATTED FILE OF CLUSTER NAME AND CLUSTER # AT DBT MATCH
        C       DATA ARE DISC FOR AA PROCESSING...LACIEY VERSION
        C       ALSO PRINTS REPORT FOR AA INTERNAL USE
        C       OUTPUT WAS OPENED IN SUBROUTINE BIASC
0003            BYTE CNAME(60,12)
0004            COMMON/CLUSTR/CNAME,CNUM
0005            NT = 3
        C       WRITE TWO RECORDS,  FIRST,THE NUMBER OF CLUSTERS, THEN, THE ALPHANUMERIC
        C       DATA...12 CHARACTERS PER CLUSTER
0006            WRITE(NT)CNUM
0007            WRITE(NT)((CNAME(J,K),K=1,12),J=1,CNUM)
0008            WRITE(6,100)((CNAME(J,K),K=1,12),J=1,CNUM)
        C       CLOSE OUTPUT FILE
0009            CALL CLOSE(NT)
0010            RETURN
0011      100   FORMAT(1H ,15X,'CLUSTER NAME      DBT NAME',//,60(18X,6A1,9X,6A1,/))
0012            END
```

3-34
36

Figure 16.— Listing for subroutine RITE7.

3.3.8 SUBROUTINE TURNON

3.3.8.1 Linkage

Subroutine TURNON is called by subroutines READH, BIASC7, and DOTS7.

3.3.8.2 Interface

TURNON interfaces with its calling routines via COMMON block NAME (see section 3.3.2.2.1) and a passed parameter, NT.

3.3.8.3 Input

There is no input to this subroutine.

3.3.8.4 Output

TURNON has no output.

3.3.8.5 Storage

This subroutine requires 162 words of storage.

3.3.8.6 Description

TURNON opens a file with the file name contained in byte array NM. If NT equals 1, the input file is opened as UNIT equals 1. If NT equals 2 to 6, an unformatted file is opened as unit NT. If NT is greater than 6, a formatted file is opened as unit (NT - 6). Prior to opening the file, the routine prints a message containing the passed unit number parameter, NT, and the file name, NM.

3.3.8.7 Flow Chart

The flow diagram for subroutine TURNON is given in figure 17.

3.3.8.8 Listing

The listing for this subroutine is given in figure 18.

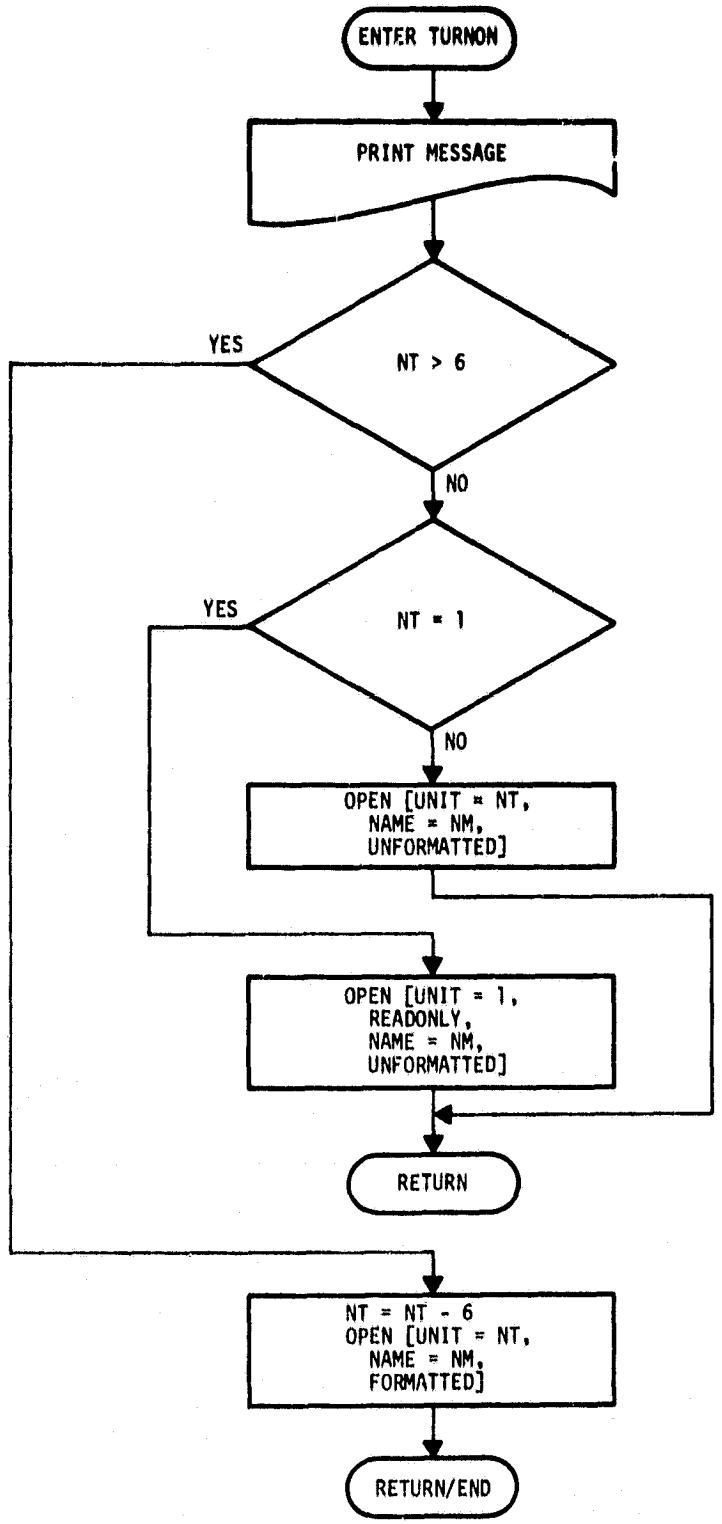

Figure 17.— Flow diagram for subroutine TURNON.

```
FORTRAN IV-PLUS V02-51          07:46:30    05-MAY-78          PAGE 1
TURNON,FTN      /I4:BLOCK3/WR

0001            SUBROUTINE TURNON(NT)
0002            IMPLICIT INTEGER(A-Z)
      C*****************************************************************
      C       ROUTINE OPENS DIRECT ACCESS FILES FOR PROGRAM CCITT6A
      C       IF UNIT NUMBER PASSED (NT) IS >6, OPEN A FORMATTED FILE
      C       ELSE OPEN AN UNFORMATTED FILE
      C       USE UNIT 1 FOR A FILE WHICH MUST BE 'OLD'
      C*****************************************************************
0003            BYTE NM(29)
0004            COMMON/NAME/NM
0005            PRINT 100,NT,(NM(K),K=1,24)
0006      100   FORMAT(1HO,'  OPENING UNIT ',I1,' ON DISC TO ACCESS FILE '
     *,24A1,/)
0007            IF(NT.GT.6) GO TO 2
0008            IF(NT.EQ.1) OPEN(UNIT=NT,NAME=NM,READONLY,
     *  TYPE='OLD',ACCESS='SEQUENTIAL',FORM='UNFORMATTED')
0009            IF(NT.EQ.1) RETURN
0010            OPEN(UNIT=NT,NAME=NM,TYPE='UNKNOWN',ACCESS='SEQUENTIAL',
     1  FORM='UNFORMATTED')
0011            RETURN
0012      2     NT = NT - 6
0013            OPEN(UNIT=NT,NAME=NM,TYPE='UNKNOWN',ACCESS='SEQUENTIAL',
     1  FORM='FORMATTED',CARRIAGE CONTROL='FORTRAN')
0014            RETURN
0015            END
```

3-37
39

Figure 18.— Listing for subroutine TURNON.

3.3.9 SUBROUTINE DOTS7

3.3.9.1 Linkage

Subroutine DOTS7 is called by CCIT7 once via the main entry and 14 times via entry DOTS1. DOTS7 calls subroutine TURNON twice and subroutines PRINT7 (via entry PRINTD) and STCOD7 once.

3.3.9.2 Interface

Subroutine DOTS7 interfaces with TURNON via COMMON block NAME (see section 3.3.2.2.1) and with CCIT7 via COMMON blocks BUF (see section 3.3.1.2.1) and DOTS (see section 3.3.4.2.1).

3.3.9.3 Input

There is no input to this subroutine.

3.3.9.4 Output

Subroutine DOTS7 writes formatted, card-image records onto two disk-based output files opened on the initial call to the routine.

3.3.9.5 Storage

This subroutine requires 976 words of storage.

3.3.9.6 Description

DOTS7 processes CCIT 'D' records into two formatted files of analyst-labeled dots. When called as DOTS7, the routine initializes the unit parameters, NT and MT, and the dot counters, KOUNT1 and KOUNT2. Then the elements of the array NM are set to name the file to receive the type 1 analyst-labeled dot data, and TURNON is called to open this file. NM(24) is redefined (1 → 2) to provide the name of the type 2 dot output file, and TURNON is called to open this file. Subroutine STCOD7 is called to obtain the two-byte parameter ST, the alphabetic state code for the segment. Control then returns to CCIT7.

When called as DOTS1, the routine processes one 720-byte 'D' record. For each analyst-labeled dot, one record is written. For type 1 dots, the data are written onto unit 2; for type 2 dots, the data are written onto unit 3. KOUNT1 is incremented for each type 1 dot, and KOUNT2 is incremented for each type 2 dot.

After processing all 209 dots (14 calls from CCIT7), a blank record is written into each output file. Then both output files are closed, and a message listing KOUNT1 and KOUNT2 is printed via a call to the PRINT7 subroutine entry PRINTD.

3.3.9.7 Flow Chart

The flow diagram for subroutine DOTS7 is given in figure 19.

3.3.9.8 Listing

The listing for this subroutine is given in figure 20.

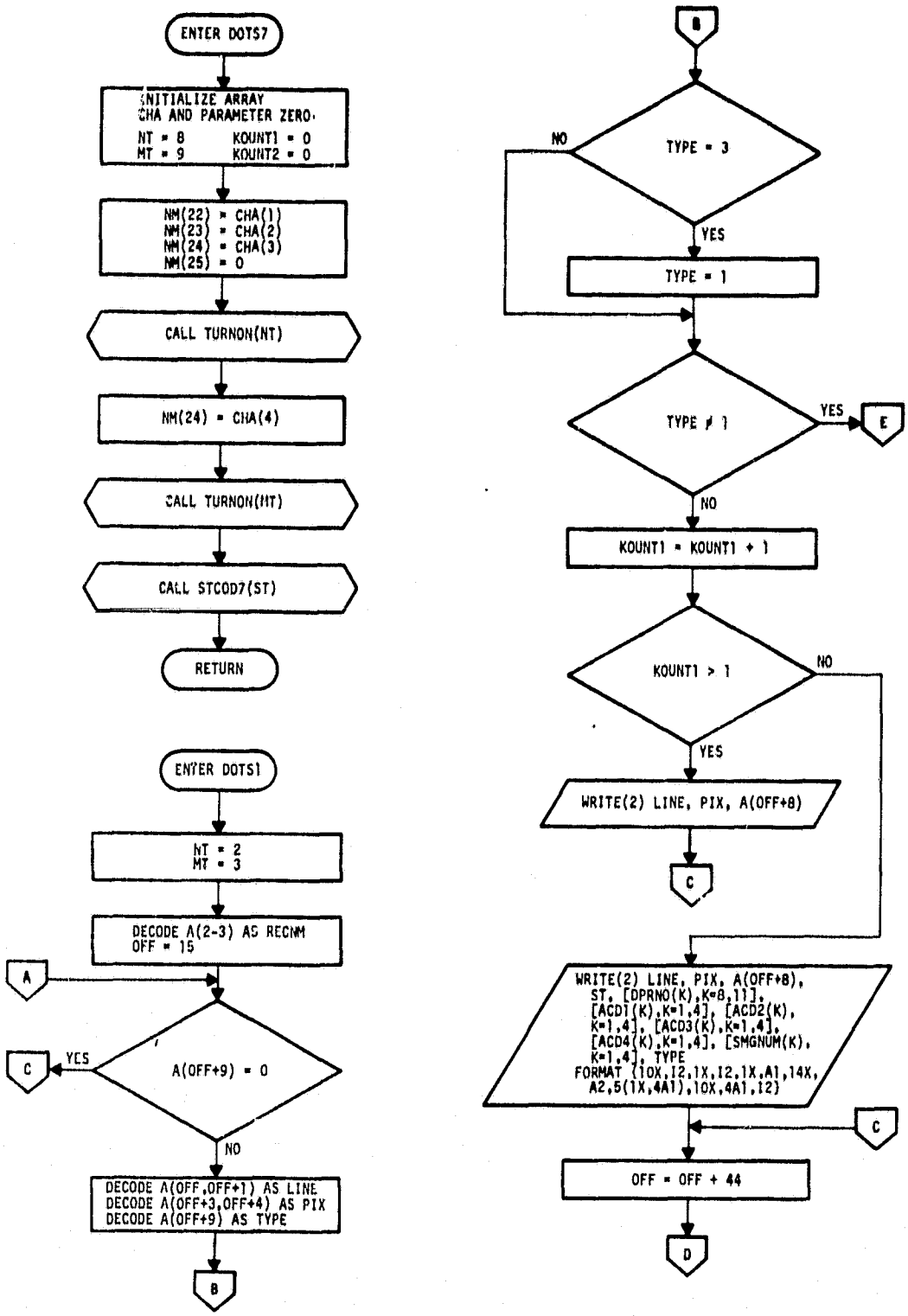

Figure 19.— Flow diagram for subroutine DOTS7.

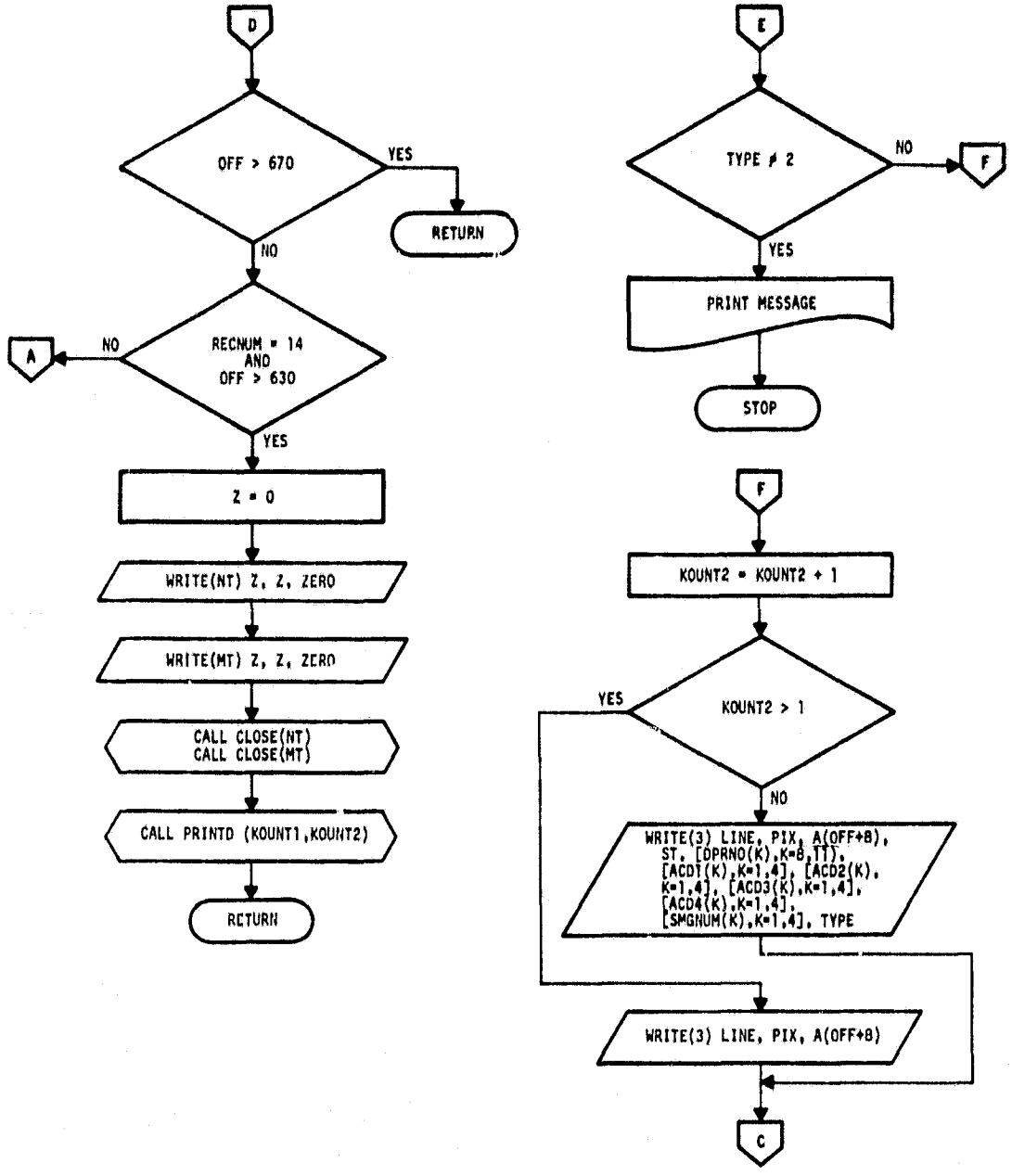

Figure 19.— Concluded.

```
DOTS7.F[illegible]   [illegible]
      SUBROUTINE DOTS7
      IMPLICIT INTEGER (A-Z)
C*************************************************************
C     [illegible] RECORDS OF [illegible] DATS, [illegible], RECORDS INTO
C     [illegible] ANALYSIS PROGRAMS SUCH AS SPATL,
C     [illegible]
C     [illegible]-7 VERSION
C     TYPE [illegible] TYPE 2 DATS AND BOTH ARE CHECKED TO
C     SEE IF THEY [illegible] LABELED. IF SO, THEY ARE WRITTEN
C     TO [illegible] FILES SSSSYY[illegible].AI1 AND SSSSYYDDD.AI2 RESPEC-
C     TIVELY.
C
C     [illegible] STCOD7 FETCHES THE STATE [illegible], ST(A3), FOR
C     THE SEGMENT FROM A TABLE SO THAT IT CAN BE INCLUDED IN
C     THE [illegible] FILES DATA. [illegible] ONLY THE FIRST LINE OF
C     DATA [illegible] EACH FILE [illegible] THE ANCILLARY DATA FOR THE
C     [illegible] CLASSIFICATION DATE.
C*************************************************************
      BYTE [illegible](25),ACD1(4),ACD2(4),ACD3(4),ACD4(4),SMGNUM(4),DPRS2(
     1 23),A(720),CHA(4),ZERO
      [illegible]
      [illegible] /B F/
      COMMON /DOTS7/ [illegible],ACD1,ACD2,ACD3,ACD4,SMGNUM
      DATA CHA/1HA,1H[illegible],1H1,1H2/
      DATA ZERO/1H0/
C     SET UNIT NUMBERS
      IT = 8
      IT = 9
C     SET COUNTERS FOR EACH CLASS OF DOT
      KOUNT1 = 0
      KOUNT2 = 0
      [illegible](22) = CHA(1)
      [illegible](23) = CHA(2)
      [illegible](24) = CHA(3)
      [illegible](25) = 0
C     OPEN THE TYPE 1 DOTS FILE
      CALL [illegible]
      [illegible](24) = CHA(4)
C     OPEN THE TYPE 2 DOTS FILE
      CALL [illegible]( T)
C     FIND ALPHABETIC STATE CODE
      CALL STCOD7(ST)
      RETURN
      ENTRY DOTS1
C     PROCESS DOT RECORDS
      IT = 2
      IT = 3
C     FIND THE CURRENT RECORD NUMBER
      DECODE(2,100,A(2))RECNM
      PFF = 15
C     IF THIS DOT HAS NOT LABELED BY THE AI'S SKIP TO THE NEXT DOT
  200 IF(A(PFF+9).EQ.ZERO) GO TO 599
C     DECODE THE LINE AND PIXEL NUMBERS OF A LABELED DOT
      DECODE(2,100,A(PFF))LINE
      DECODE(2,100,A(PFF+3))PIX
C     FIND OUT WHAT TYPE DOT IT IS, A TYPE OF '3' MEANS A STARTING VECTOR
```

Figure 20.— Listing for subroutine DOTS7.

ORIGINAL PAGE IS
OF POOR QUALITY

```
FORTRAN IV-PLUS V02-01          12:31:26     19-SEP-79          PAGE 2
DETS7.FT            /TR:BLOCKS/WR

      C       TYPE 1 DOT S' RELABEL AS TYPE 1
              RECORD(1,110,A(OFF+9))TYPE
              IF(TYPE.EQ.3)TYPE=1
              IF(TYPE.NE.1)GO TO 300
      C       WRITE OUT TYPE 1 DOTS
              KOUNT1 = KOUNT1 + 1
              IF(KOUNT1.GT.1)GO TO 700
              WRITE(NT,140)LINE,PIX,A(OFF+8),ST,(NN(K),K=16,20),
            1 (ACD1(K),K=1,4),(ACD2(K),K=1,4),(ACD3(K),K=1,4),
            2 (ACD4(K),K=1,4),(SPGNUM(K),K=1,4),TYPE
              GO TO 701
        700   WRITE(NT,140)LINE,PIX,A(OFF+8)
        701   GO TO 999
        300   IF(TYPE.NE.2)GO TO 666
      C       WRITE OUT TYPE 2 DOTS
              KOUNT2=KOUNT2 + 1
              IF(KOUNT2.GT.1) GO TO 710
              WRITE(NT,140)LINE,PIX,A(OFF+8),ST,(NN(K),K=16,20),
            1 (ACD1(K),K=1,4),(ACD2(K),K=1,4),(ACD3(K),K=1,4),(ACD4(K),K=1,
            2 4),(SPGNUM(K),K=1,4),TYPE
              GO TO 999
        710   WRITE(NT,140)LINE,PIX,A(OFF+8)
        999   OFF = OFF + 44
              IF(OFF.GT.670) RETURN
              IF(RECNM.EQ.14.AND.OFF.GT.630) GO TO 888
              GO TO 200
      C       DONE - WRITE BLANK RECORDS CLOSE BOTH OUTPUT FILES AND PRINT
        888   Z = 0
              WRITE(NT,140) Z,Z,ZERO
              WRITE(MT,140) Z,Z,ZERO
              CALL CLOSE(NT)
              CALL CLOSE(MT)
              CALL PRINTD(KOUNT1,KOUNT2)
              RETURN
      C       WRONG TYPE DOT: WRITE ERROR MESSAGE
        666   PRINT 120,A(OFF+9)
              STOP
        120   FORMAT(1H ,' ERROR IN DETS...IMPROPER TYPE ',A1,//)
        100   FORMAT(I2)
        110   FORMAT(I1)
        140   FORMAT(10X,I2,1X,I2,1X,A1,14X,A2,5(1X,4A1),10X,4A1,I2)
              END
```

Figure 20.— Concluded.

3.3.10 SUBROUTINE STCOD7

3.3.10.1 Linkage

STCOD7 is called once by subroutine DOTS7.

3.3.10.2 Interface

STCOD7 interfaces with DOTS7 via COMMON block FNAME (see section 3.3.1.2.2) and passed parameter ST.

3.3.10.3 Input

There is no input to this subroutine.

3.3.10.4 Output

Subroutine STCOD7 has no output.

3.3.10.5 Storage

This subroutine requires 1099 words of storage.

3.3.10.6 Description

STCOD7 locates the correct two-character alphabetic state code, ST, for a given segment number via table lookup. Note: The table given is only valid for AA LACIE U.S. Great Plains blind sites for the 1978 Transition Year.

3.3.10.7 Flow Chart

The flow diagram for subroutine STCOD7 is given in figure 21.

3.3.10.8 Listing

The listing for this subroutine is given in figure 22.

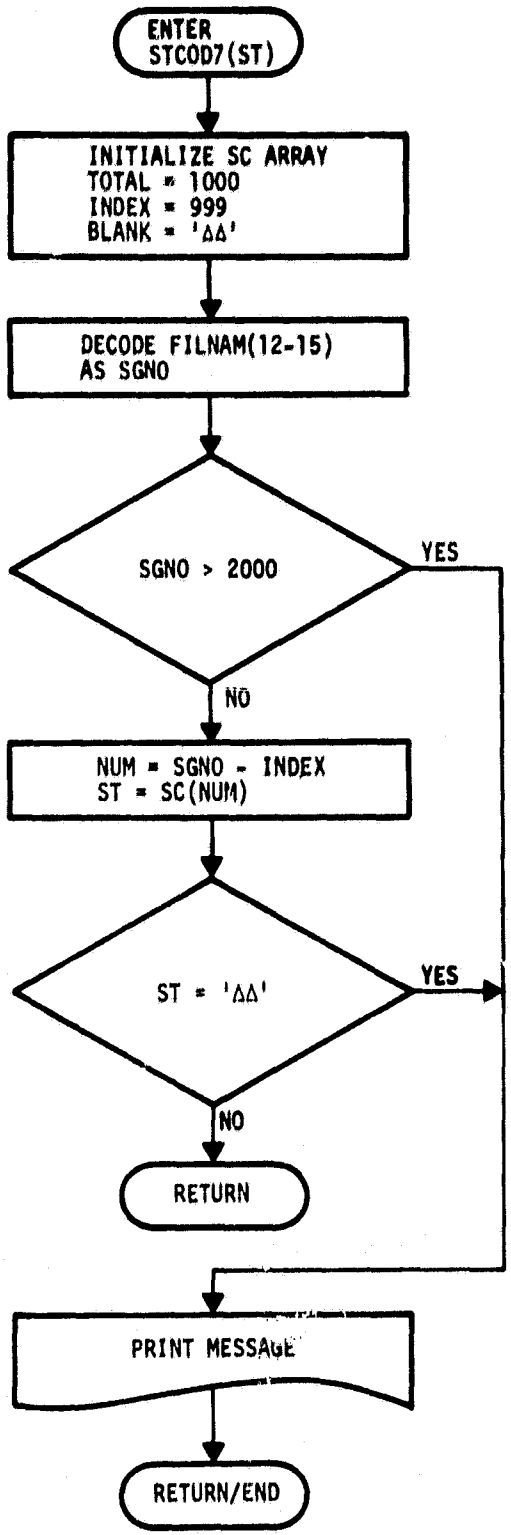

Figure 21.— Flow diagram for subroutine STCOD7.

```
FORTRAN IV-PLUS V32-51          13:50:54    25-JUN-79          PAGE 1
STC2D7.FTN      /TR:ALECKS/aR

0001            SUBROUTINE STC2D7(ST)
0002            IMPLICIT INTEGER(A-Z)
        C       FIND THE TWO LETTER STATE CODE FOR THE LACIE TY BLIND SITE
        C       SPECIFIED IN THE FILENAME
0003            DIMENSION SC(1000)
0004            BYTE FILNAM(24)
0005            COMMON/FNAME/FILNAM,SKIP
0006            DATA SC/1000*2H  /
0007            DATA TOTAL/1000/
0008            DATA INDEX/999/
0009            DATA BLANK/2H  /
0010            DATA SC(4),SC(25),SC(28),SC(65),SC(70),SC(92),SC(347),SC(503),
               1 SC(510),SC(851),SC(858)/40*2HCA/
0011            DATA SC(382),SC(919),SC(519),SC(925),SC(967),SC(826),SC(836),SC(843),
               1 SC(844),SC(846),SC(888)/11*2HNV/
0012            DATA SC(10),SC(74),SC(156),SC(201),SC(582),SC(583),SC(493),SC(511),SC(477),
               1 SC(573),SC(584),SC(592),SC(595),SC(597),SC(878)/19*2HNB/
0013            DATA SC(104),SC(390),SC(530),SC(943),SC(969),SC(991),SC(996),SC(988),
               1 SC(726),SC(732),SC(743),SC(936),SC(943),SC(946),SC(949)/19*2HNT/
0014            DATA SC(24),SC(36),SC(42),SC(48),SC(98),SC(92),SC(197),SC(166),
               1 SC(174),SC(176),SC(178),SC(179),SC(224),SC(287),SC(288),SC(282),
               2 SC(289),SC(287),SC(294),SC(886),SC(3501),SC(3421),SC(3981),SC(3991),
               3 SC(278),SC(857),SC(862),SC(864),SC(877),SC(879),SC(881),SC(884),
               4 SC(889),SC(891),SC(892)/35*2HWS/
0015            DATA SC(56),SC(78),SC(79),SC(85),SC(89),SC(265),SC(271),SC(273),
               1 SC(274),SC(333),SC(379),SC(376),SC(378),SC(692),SC(887)/19*2HTX/
0016            DATA SC(155),SC(48?),SC(589),SC(688),SC(689),SC(677),SC(679),SC(698),
               1 SC(796),SC(795),SC(803),SC(818),SC(131),SC(614)/14*2HOR/
0017            DATA SC(91),SC(220),SC(222),SC(227),SC(232),SC(237),SC(239),SC(243),
               1 SC(294),SC(398),SC(398),SC(366),SC(389),SC(489),SC(598),SC(598),
               2 SC(666),SC(671),SC(672),SC(696)/28*2HOK/
0018            DATA SC(388),SC(393),SC(395),SC(488),SC(458),SC(468),SC(482),SC(493),
               1 SC(474),SC(572),SC(585),SC(603),SC(612),SC(613),SC(618),SC(628),
               2 SC(628),SC(631),SC(637),SC(644),SC(691),SC(694),SC(697),SC(699),
               3 SC(662),SC(665),SC(910),SC(918),SC(919),SC(921),SC(925)/31*2HNB/
0019            DECODE(4,200,FILNAM(12))SGN8
0020            IF(SGN8.GE.2000) GO TO 900
0021            NUM = SGN8 + INDEX
0022            ST = SC(NUM)
0023            IF(ST.EQ.BLANK) GO TO 900
0024            RETURN
0025      900   PRINT 100,SGN8
0026      100   FORMAT(1H ,10X,'NO STATE CODE ASSIGNED FOR SEGMENT',I5,/)
0027      200   FORMAT(I4)
0028            RETURN
0029            END
```

Figure 22.— Listing for subroutine STC0D7.

3.3.11 SUBROUTINE PRINT7

3.3.11.1 Linkage

PRINT7 is called by CCIT7 via entries PRINT7 and PRINTE, by HEADER via entry PRINTH, by DOTS via entry PRINTD, and by CLUST7 via entry PRINTC. All other called routines are Image Processor system routines.

3.3.11.2 Interface

PRINT7 interfaces with HEADER via COMMON block DOTS (see section 3.3.4.2.1), with CCIT7 via COMMON block FNAME (see section 3.3.1.2.2), with DOTS7 via passed parameters K1 and K2, and with CLUST7 via passed parameter RCNUM.

3.3.11.3 Input

There is no input to this subroutine.

3.3.11.4 Output

PRINT7 prints messages on the line printer.

3.3.11.5 Storage

This subroutine requires 783 words of storage.

3.3.11.6 Description

PRINT7 provides most line printer message output for the CCIT7 processor. This output provides a processing record for AA status and tracking activity. The routine uses system routines TIME and DATE to obtain data for header and trailer line printer messages for each run.

3.3.11.7 Flow Chart

The flow diagram for subroutine PRINT7 is given in figure 23.

3.3.11.8 Listing

The program listing for this subroutine is given figure 24.

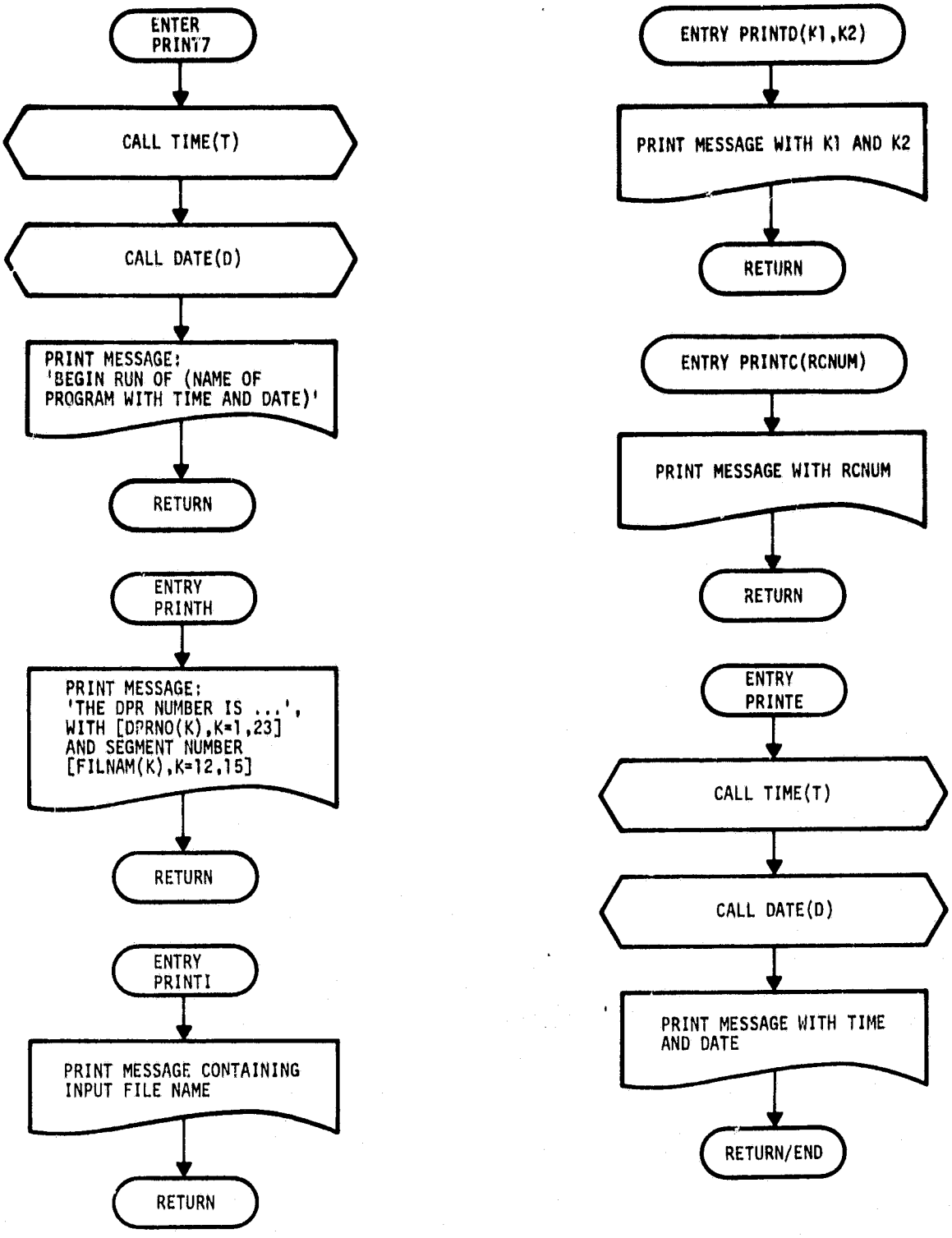

Figure 23.— Flow diagram for subroutine PRINT7.

```
FORTRAN IV+PLUS V02-51          14:13:29    SC=_UN=78          PAGE 1
PRINT7.FTN      /TR:BLOCKS/WR

0001            SUBROUTINE PRINT7
0002            IMPLICIT INTEGER(A-Z)
        C       MASTER SUBROUTINE .E PRINT HEADERS AND DATA OUTPUTS FOR CCIT
        C       PROCESSOR PROGRAM...LACIE) VERSION
0003            BYTE A(720),FILNAM(24),DPRNB(23),ACD1(4),ACD2(4),ACD3(4),
               *ACD4(4),SMGNUM(4),T(8),D(9)
0004            COMMON/FNAME/FILNAM,SKIP
0005            COMMON/DATS/DPRNB,ACD1,ACD2,ACD3,ACD4,SMGNUM
0006            COMMON/BUF/A
0007            CALL TIME(T)
0008            CALL DATE(D)
0009            PRINT 100,D,T
0010    100     FORMAT(1H1,'   BEGIN RUN OF CCIT7 PROCESSOR PROGRAM',//,
               *10X,'ON  ',9A1,'  AT  ',8A1)
0011            RETURN
0012            ENTRY PRINTH
0013            PRINT 500,(DPRNB(K),K=1,23),(FILNAM(K),K=12,15)
0014    500     FORMAT(1H ,' THE DPR NUMBER IS ',23A1,' FOR SEGMENT NUMBER',
               *1X,4A1)
0015            RETURN
0016            ENTRY PRINTI
0017            PRINT 600,(FILNAM(K),K=1,24)
0018    600     FORMAT(1H ,' PROGRAM CCIT7',//,1H ,10X,'ON FILE ',24A1,//)
0019            RETURN
0020            ENTRY PRINTD(K1,K2)
0021            PRINT 700,K1,K2
0022    700     FORMAT(1H ,10X,' THERE WERE',I3,' TYPE 1, AND',
               *I3,' TYPE 2 DBTS ASSIGNED')
0023            RETURN
0024            ENTRY PRINTC(RCNUM)
0025            PRINT 800,RCNUM
0026    800     FORMAT(1H ,10X,'PROCESSING',I3,' CLUSTERS FOR DBT MATCH')
0027            RETURN
0028            ENTRY PRINTE
0029            CALL TIME(T)
0030            CALL DATE(D)
0031            PRINT 200,D,T
0032            PRINT 200,D,T
0033    200     FORMAT(1H ,'   END RUN OF CCIT7 PROCESSOR PROGRAM',//,10X,
               *'ON  ',9A1,'  AT  ',8A1)
0034            RETURN
0035            END
```

3-49

ORIGINAL PAGE IS OF POOR QUALITY

Figure 24.— Listing for subroutine PRINT7.

4. OPERATIONS

This section presents all the information necessary to obtain proper execution of the CCIT7 processor program.

4.1 OPERATORS GUIDE

This section explains the system hardware configuration and execution (run) setup for the CCIT7.

4.1.1 HARDWARE CONFIGURATION

The nominal configuration is the Earth Observations Division/Data Techniques Laboratory (EOD/DTL) PDP 11/45 processor with the RSX 11-D operating system. The system must have the input CCIT files resident on either the system disk or a user disk. The output files are written onto the same disk and under the same user identification code (UIC) as the resident input data. The input files are created using program AACCIT, described in JSC-13893. (See section 2 of this specification.)

4.1.2 PROGRAM EXECUTION

4.1.2.1 INTERACTIVE SETUP

a. Edit file CCIT7.DAT for the proper file name and the value of parameter SKIP (24A1,I2). The file name takes the form:

DBX:[abc,d]SSSSYYDDD.wxy

where

X = Disk unit number

SSSSYYDDD = Input file name

wxy = Input file type; i.e., .CCO

[abc,d] = UIC for the input file

b. Mount the proper disk pack on the drive.

c. Type 'RUN CCIT7'.

d. When message CCIT7-STOP appears on the monitor, collect a single-page report at the line printer, and check the listing to ensure that the ending message was printed and that the various steps were properly executed.

4.1.2.2 BATCH SETUP

a. Prepare a batch run request detailing the disk configuration required.

b. Set up a batch run deck as in table 2. The required steps follow:

- Delete CCIT7.DAT.
- Create CCIT7.DAT with card images, as given in section 4.1.2.1.
- Run CCIT7.TSK.

4.2 USERS GUIDE

The CCIT7 program is designed to obtain a small fraction of the data from a CCIT disk file and to reformat these data into a form directly used by several AA software modules. This program will not execute correctly for CCIT's other than those created under LACIE version 7. The approximate dates of valid CCIT's for version 7 are 77305 through the date of issuance of this specification.

4.3 MAINTENANCE DOCUMENTATION

Not applicable.

TABLE 2.— BATCH RUN DECK SETUP

```
$JOB/NAME=AA/MCR/LIMIT=99/ACCOUNT=1106
$MCR PIP
CCIT7.DAT;*/DE
$CREATE CCIT7.DAT
   :
Card images for file name and SKIP parameter (24A1,I2)
   :
Blank card
$EOD
$MCR REM RSXBAT
$RUN CCIT7.TSK
$EOJ
```

APPENDIX

FORMAT OF .CLO FILE

APPENDIX

FORMAT OF .CLO FILE

Record 1: The first record contains a single integer representing the number of classes detailed in the next record. Sixteen bytes of data follow for each class (minimum of 2 classes, maximum of 26 classes).

Record 2:

Byte number	Data description (ASCII)
1	Class 1 label (W, S, G, N, etc.)
2-6	Pixel population; PPPPP
7-9	Uncorrected proportion; M.MM (implied decimal point)
10-12	Corrected proportion; N.NN (implied decimal point)
13-16	Variance; .VVVV (implied decimal point)
17	Class 2 label (W, S, G, N, etc.)
18-22	Pixel population; PPPPP
23-25	Uncorrected proportion; M.MM (implied decimal point)
26-28	Corrected proportion; N.NN (implied decimal point)
29-32	Variance; .VVVV (implied decimal point)
⋮	⋮

Record 3: This record contains a single integer giving the number of clusters in the classification, CNUM.

Record 4: The fourth record contains 12 bytes of ASCII character data for each cluster; e.g., 12*CNUM bytes of data. The first 6 bytes of each group of 12 are the cluster label; e.g., NOCL17. The last six bytes of each group are the identity of the dot used to label the clusters; e.g., DOT103. Only type 1 dots are used to label clusters.